A Hybrid Elastic Model allowing Real-Time Cutting, Deformations and Force-Feedback for Surgery Training and Simulation

Stéphane Cotin, Hervé Delingette, Nicholas Ayache

▶ **To cite this version:**

Stéphane Cotin, Hervé Delingette, Nicholas Ayache. A Hybrid Elastic Model allowing Real-Time Cutting, Deformations and Force-Feedback for Surgery Training and Simulation. The Visual Computer, 2000, 16 (8), pp.437–452. 10.1007/PL00007215 . inria-00615105

HAL Id: inria-00615105
https://inria.hal.science/inria-00615105

Submitted on 17 Aug 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Published in the "Visual Computer" journal, Vol 16, No 8, pages 437-452, 2000

A Hybrid Elastic Model allowing Real-Time Cutting, Deformations and Force-Feedback for Surgery Training and Simulation

Stéphane Cotin, Hervé Delingette, Nicholas Ayache
INRIA Sophia Antipolis
Projet Epidaure
2004 Route des Lucioles
06902 Sophia-Antipolis, BP 93, France
E-mail {scotin,hdeling,na}@sophia.inria.fr

Keywords: Surgery Simulation, Finite Elements, Linear Elasticity, Spring-Mass models, Soft Tissue Modeling.

Abstract

We propose three different physical models based on linear elasticity theory and finite-elements modeling that are well-suited for surgery simulation. The first model combines pre-computed deformations to deform in real-time large size meshes but does not allow any topological changes to the mesh. The second model is similar to the spring-mass models where volumetric deformations and cutting operations can be simulated on small size meshes in real-time. Finally, we have developped a third method combining the previous two solutions into a hybrid model thus allowing the simulation of deformations and cutting on complex anatomical structures.

1 Introduction

1.1 Minimally Invasive Surgery Simulation

The most recent major advance in the craft of abdominal surgery was the development of laparoscopic surgery. In this type of surgery, abdominal operations such as hepatic surgery are accomplished through small incisions rather than a large one that might be a foot long. The abdomen is blown up with gas so that there is open space inside. A video camera is introduced into the abdomen through one of the small incisions. The video image is magnified and transmitted to a high resolution monitor, allowing the surgeon to see the abdominal anatomy with great clarity.

If minimally invasive surgery drastically reduces the patient morbidity, it also significantly degrades the surgeon access to the patient body. More precisely, the lack of tridimensional perception and the necessity for a new and specific hand-eye coordination are the main impediments that a video-surgeon should overcome. In this scope, there is a large interest in developing video-surgery simulation software for providing a comprehensive gesture training system. The growing interest regarding surgical simulators is also related to the new perspectives offered by the increase of computer power. Although this is an important factor in the development of a real-time simulation system, the current performances of computers[1] does not allow for real-time realistic[2] simulations: the development of new algorithms is still mandatory.

There are several key problems in the development of a surgical simulator (Ayache et al., 1998). First of all, geometric and physical models of anatomical structures must be defined. The geometry can be obtained from various medical image modalities, while the deformable nature of the soft tissues are measured through biomechanical studies. Furthermore, all computation must be performed fast enough to sustain real-time interactions. In the framework of surgical simulation, real-time means that visual and haptic feedback can be reproduced at a correct frequency. To support the user with visual feedback, a deformable model has to be updated in less than 20 ms. To support haptic feedback, the computation time must typically be ten times smaller. Being able to provide surgical realism at interactive rates of simulation is the most challenging problem in surgical simulation.

1.2 State-of-the-art in Soft Tissue Modeling

The pioneering work of Terzopoulos (Terzopoulos et al., 1987), Waters (Waters, 1992) and Platt (Platt and Barr, 1988) have shown the advantages of physically-based models over previous com-

[1] We do not consider the use of super computers or parallel machines.

[2] By "realistic simulation" we mean that the perceptual information transmitted to the surgical trainee is accurate to avoid the introduction of any incorrectness in the training process.

puter animation techniques. In the area of surgery simulation, a large variety of these models have been implemented as described in a survey on this topic (Delingette, 1998). For instance, great interest has been given to spring-mass models due to their simplicity of implementation and their relatively low computational complexity (Baumann and Glauser, 1996; Meseure and Chaillou, 1997; Kuehnapfel and Neisius, 1993). Kuehnapfel and Neisius (Kuehnapfel and Neisius, 1993) present a simulation of endoscopic surgery based on a surface spring-mass model. Although in this case the interactions are driven by instruments with motion sensors, no force feedback is provided. The "chain-mail" system developed by Gibson *et al.* (Gibson et al., 1997) takes into account the volumetric nature of organs with a deformation law derived from a spring-mass model. This approach is computationally very efficient, but it is not well-suited for interactive visual display and for identifying material parameters.

On the other hand, several authors have based their soft tissue models on continuum mechanics theory, and the use of elastic solids is widely described in the literature (Bainville et al., 1995; Song and Reddy, 1995; Chen and Zeltzer, 1992; Speeter, 1992). Finite element methods are often considered to be less efficient than spring-mass models. However, there is a growing trend in using finite element soft tissue models for real-time computation, as shown for instance by Szekely *et al.* (Szekely et al., 1998) who simulate the deformation of a non-linear elastic material using a parallel processing architecture.

Several modifications and simplifications may be introduced to reduce computation time. In particular, linear elasticity has often been used as a trade-off between biomechanical realism and real-time computation. In (Bainville et al., 1995), Bainville defines the evolution of a set of rigid and deformable solids under the influence of various forces. In this case, the deformation law is represented by a hyper-elastic quasi-static model, associated to a finite element method for the numerical resolution. Unfortunately, the computation times makes this approach unpractical for real-time surgical simulation.

Also, in (Bro-Nielsen and Cotin, 1996), Bro-Nielsen *et al.* use a condensation technique (Zienkiewicz, 1977) in order to reduce the computation time in the deformation process of a linear elastic material. With such an approach, the computation time required for the deformation of a volumetric model can be reduced to the computation time of a model only involving its surface nodes. In (Cotin et al., 1999), we have also proposed a method for real-time interaction with a volumetric deformable model of an organ. This method, based on a set of pre-computed equilibrium solutions is very efficient but does not allow any topology changes, as needed to simulate tissue cutting.

Indeed, methods based on stiffness matrix inverse pre-computation cannot be used to simulate soft tissue cutting or tearing, since the calculation cost of updating these matrices is too high (Cotin, 1997). Few authors have proposed a method for simulating tissue cutting in the

framework of surgery simulation. Song and Reddy (Song and Reddy, 1995) have described a technique for cutting linear elastic objects defined as finite element models. However, this technique was only applied to simple two dimensional objects. Another general approach used by several researchers consists in using spring-mass models. By construction, the underlying geometry of these models can easily be modified to represent topology changes. However, spring-mass models are discrete representations of a continuum and the update of stiffness and mass values – in order to have a realistic behavior after a cutting operation – is hard to handle.

1.3 Overview of the proposed approach

In this paper, we propose a new method for deforming and cutting soft tissue models with a large number of vertices. Our approach is based on two separate soft tissue models: the first model, previously described in (Cotin et al., 1999), can be deformed in an extremely efficient manner (less than 3ms of computation time for our liver model). Then we introduce a new soft tissue model, called tensor-mass model, that is defined as a linear elastic dynamic mesh that can be cut and deformed with a computational complexity linear in the number of nodes. This model is as simple to implement and as efficient as spring-mass models, but it is based on continuum mechanics and linear elasticity theory. Its compact data structure makes it particularly well-suited for the simulation of tearing and cutting. Finally, in order to use large soft tissue models, we propose to combine these two approaches into an hybrid elastic model.

2 Linear elasticity

The physical behavior of a soft tissue model may be considered as linear elastic if the displacements applied to it remain small (Fung, 1993; Maurel et al., 1998) (less than 10% of the typical object size); as the displacements increase, the linear elastic approximation becomes less valid. Although it is a major drawback of linear elasticity, the integration of force feedback in the simulation loop allows us to control the range of deformation. When the surgical trainee deforms the virtual organ, the force applied in its hand will increase proportionally to the deformation, thus preventing large deformations. Consequently, the displacements remain reasonably small. Another interest of linear elasticity is the possibility to compute any mesh deformation from the knowledge of a finite set of elementary deformations, as we will see in the next section.

First, let's define a reference volumetric anatomical model $\mathcal{M}_{\text{initial}}$ corresponding to the organ in its rest position. Under external constraints, for instance a surgical instrument, the anatomical model $\mathcal{M}_{\text{initial}}$ will deform. We represent this deformation – the difference between the current shape and the rest shape – by a *displacement vector* $\mathbf{U}(x, y, z)$ for $(x, y, z) \in \mathcal{M}_{\text{initial}}$ and we write $\mathcal{M}_{\text{deformed}} = \mathcal{M}_{\text{initial}} + \mathbf{U}(x, y, z)$. The displacement vector $\mathbf{U}(x, y, z)$ has three

components:

$$\mathbf{U}(x,y,z) = \begin{cases} u(x,y,z) \\ v(x,y,z) \\ w(x,y,z) \end{cases}$$

The displacement vector $\mathbf{U}(x,y,z)$ does not characterize the deformation of the anatomical model. For instance, under a translation $\mathbf{T}$ of the model $\mathcal{M}$, the displacement vector is $\mathbf{U}(x,y,z) = \mathbf{T}$, but the model does not undergo any deformation. For a linear elastic material, the elastic energy $W_{Elastic}$ measuring the amount of deformation of $\mathcal{M}_{\text{deformed}}$, is defined as (Ciarlet, 1987):

$$W_{\text{Elastic}} = \frac{\lambda}{2}(trE)^2 + \mu trE^2 \tag{1}$$

where

- the 3×3 symmetric matrix E (known as the Green-St Venant strain tensor) is defined as:

$$E = \frac{1}{2}(\nabla \mathbf{U} + \nabla \mathbf{U}^T) \tag{2}$$

- λ and μ are the Lamé coefficients characterizing the stiffness of a material.

Equation (1), known as *Hooke's law*, shows that the elastic energy of a deformable object is a quadratic function of the displacement vector gradient.

In the following sections, we will place us in the framework of the finite element method and assume that the object is represented by a conformal tetrahedral mesh. Then we can compute, at each node i, a force $\mathbf{F}_i$ corresponding to the derivation of the elastic energy with respect to the node position $\mathbf{P}_i$:

$$\mathbf{F}_i = \frac{\partial W_{\text{Elastic}}}{\partial \mathbf{P}_i} \tag{3}$$

Because the elastic energy is quadratic with respect to the gradient of the displacement vector, forces $\mathbf{F}_i$ are linear functions of the displacement vectors of each node $\mathbf{P}_j$. By considering the principle of least action, the state of equilibrium of the finite element model is reached when the elastic energy is minimum. Furthermore, it can be shown that minimizing the elastic energy of a three-dimensional object requires the solution of a linear system of the form:

$$[\mathbf{K}]\mathbf{u} = \mathbf{f}$$

where :

- $[\mathbf{K}]$ is the *rigidity* matrix describing both the topology and stiffness of the discrete representation of the object,
- $\mathbf{u}$ is a vector that represents the displacement of all nodes,
- $\mathbf{f}$ is a vector that combines all external forces and boundary conditions.

In sections (3) and (4) we study successively a *quasi-static* and a *dynamic* model as a way to compute the deformation of an elastic object and the forces resulting of this deformation. Finally, we show in section (5) an hybrid approach, well suited for surgery simulation, which combines the advantages of each model.

3 Quasi-Static pre-computed Linear Elastic Model

As stated in the introduction, surgical simulation requires visual feedback and, eventually, force feedback, i.e. an update frequency up to 50Hz for the display and 300Hz for the forces[3]. When solving a problem of linear elasticity with a finite element method, the number of mesh vertices has a direct impact on the size of the matrices involved in the linear system $[\mathbf{K}]\mathbf{u} = \mathbf{f}$. This implies that even using more powerful computers, only deformable models with a small number – several hundreds – of vertices could be simulated. Most anatomical structures have a rather complex geometry and cannot be accurately described with such a limited number of vertices.

In order to speed up the interaction rate, we take advantage of the following inherent properties of linear elastic materials: the linearity and the superposition principle. We give here only a very general description of the method, an extensive description can be found in (Cotin et al., 1999). We first introduce a volumetric deformable model with the following properties:

1. The biomechanical properties of the model are approximated by a linear elastic law.
2. This model deforms under some boundary conditions expressed in terms of constrained displacements and forces.
3. The model evolves in a quasi-static state: the position of the model at time $t + 1$ is the solution of the static problem with boundary conditions given at time t. This assumption of quasi static evolution, made in many situations (Bainville, 1996; Kaiss, M. and Le Tallec, P., 1996; Bro-Nielsen, 1996; Gourret et al., 1989), considers as negligeable the effect of the acceleration and velocity in the computation of the deformation. This assumption has several advantages, in particular a simplification of the problem to be solved but also a suppression of the oscillations in the vicinity of the equilibrium as well as a reduction of the complexity of the processing of the contacts between objects. However, it is unable to exhibit any visco-elastic behavior since inertia is not taken into account. Consequently, the soft tissue model immediately reaches its rest position when all the external forces are released, without exhibiting any oscillations.

[3]The update frequency for haptic feedback depends on the stiffness of the object that is manipulated: the stiffer the object, the higher the frequency.

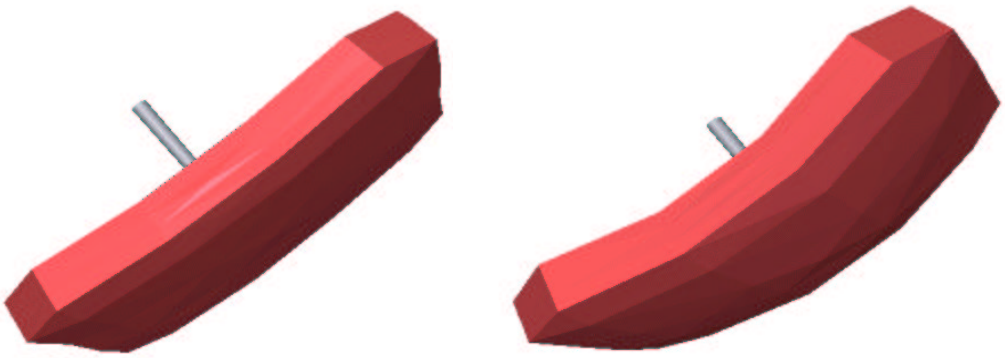


Figure 1: Deformation of plate represented as a pre-computed linear elastic model : (left) initial position (right) deformed position after imposing a displacement at the contact point with the stick.

Considering the above statements, interactive rates of deformation can be obtained in a two-step process:

1. A pre-processing stage is used to compute a set of elementary deformations of the model.
2. Any deformation can be computed in real-time as a linear combination of the pre-computed set of elementary deformations. The computational complexity is linear in the number of *surface* nodes.

We use an iterative method – conjugate gradient with or without preconditioning – to solve each linear system. During the simulation, very limited computations are required to get the exact deformation of the anatomical object. An update rate of 500Hz has been reached with a mesh having nearly 8000 tetrahedra on a 233 Mhz DEC Alphastation. We have developed a first prototype of an hepatic surgery simulator (Cotin et al., 1996; Ayache et al., 1997) including a force feedback device, based on this linear elastic soft tissue model.

The pre-processing stage can take between a few minutes and several hours depending on the size of the model and on the desired level of accuracy. This is not a problem since the result of the pre-processing stage can be saved for further simulations and therefore needs to be performed only once for a given organ model.

4 Dynamic linear elastic model or *tensor-mass* system

4.1 Motivations

The previous approach is not suited to the simulation of cutting or tearing which requires to modify the model stiffness matrix. Indeed, to keep a realistic deformation model, the pre-computed

elementary deformations are only valid for a given configuration of the stiffness matrix. When this matrix is modified, it is necessary to recompute these elementary deformations through a new pre-processing stage which is far too computationnally intensive for real-time interaction even with meshes having a few hundred nodes.

An alternative we propose in this section is to consider the evolution of a physical dynamic model which can be approximately solved by an efficient real-time iterative approach. This approach allows dealing with meshes of reasonable size – more than 2000 nodes for a computation time less than 50 ms and therefore a frame rate of 20Hz – , and provides a good approximation to the exact static equation.

The physical model is based on the Newtonian law of motion of each mesh point $\mathbf{P}_i$:

$$m_i \frac{d^2\mathbf{P}_i}{dt^2} = \gamma_i \frac{d\mathbf{P}_i}{dt} + \mathbf{F}_i \tag{4}$$

where $\mathbf{F}_i$ is obtained from the derivation of the elastic energy given by equation (3).

In the sequel of this section, we discuss the computation of $\mathbf{F}_i$ in the framework of the finite element method and numerical integration, followed by a comparison with the spring-mass formulation and a presentation of simulation results.

4.2 Definition of a dynamic linear elastic finite element model

We assume that we have a conformal tetrahedral mesh – as defined in finite element theory – describing the geometry of an anatomical structure. We denote $\mathcal{M}_{\text{initial}}$ the mesh at its rest position and $\mathbf{P}_i^0$ the initial position of each vertex. We denote $\mathbf{P}_i$ the vertex position of a deformed mesh $\mathcal{M}_{\text{deformed}}$. The computation of the linear elastic force acting on each vertex can be decomposed into three steps:

1. We first define the interpolation equations that give the displacement vector at a point (x, y, z) inside a tetrahedron T_i as a function of the 4 displacement vectors at each vertex.
2. We write the elastic energy of a tetrahedron as a function of these 4 displacement vectors.
3. We compute the elastic force $\mathbf{F}_i$ applied to the vertex $\mathbf{P}_i$

4.2.1 Displacement Vector equation

Given a deformed model $\mathcal{M}_{\text{deformed}}$, we define the *displacement vector* for each point of the mesh by linearly interpolating the displacement $\mathbf{P}_i^0\mathbf{P}_i$ of the vertices inside each tetrahedron. This amounts to choosing a linear finite element on each tetrahedron with C^0 continuity of the displacement vector across the domain.

More precisely, if we write T_i the tetrahedron defined by the four vertices $\mathbf{P}^0_{T_i(j)}$, $j = 0, \ldots, 3$, in their **rest position**, then the vector displacement at a given point $\mathbf{x} = (x, y, z)$ is defined as:

$$\mathbf{U}_{T_i}(\mathbf{x}) = \sum_{j=0}^{3} \alpha_j^{T_i}(\mathbf{x}) \mathbf{P}^0_{T_i(j)} \mathbf{P}_{T_i(j)}$$

where $\alpha_j^{T_i}(\mathbf{x})$ are the barycentric coordinates of the point $\mathbf{x}$ inside T_i. In Appendix A, we show that the shape functions $\alpha_j^{T_i}(\mathbf{x})$ are linear with respect to the position $\mathbf{x}$ and that its position derivative has a simple geometric meaning:

$$\frac{\partial \alpha_j^{T_i}}{\partial \mathbf{x}} = -\frac{\mathbf{M}_j^{T_i}}{6V(T_i)}$$

where $M_j^{T_i}$ is the normal vector of the j^{th} triangle in T_i and $V(T_i)$ is the volume of T_i. The strain tensor $E(\mathbf{x})$, defined in equation (2), is constant inside each tetrahedron since it is related to the derivatives of the displacement vector.

4.2.2 Elastic energy

If we associate with each tetrahedron T_i its linear elastic properties, i.e. the two Lamé coefficients λ_i and μ_i then, using equation (1), we can express the elastic energy $W_{Elastic}(T_i)$ of the tetrahedron T_i as a quadratic function of the coordinates of $\{\mathbf{P}_{T_i(j)}\}$ (see appendix B for more details). The total elastic energy $W_{Elastic}(\mathcal{M}_{\text{deformed}})$ required to deform $\mathcal{M}_{\text{initial}}$ into $\mathcal{M}_{\text{deformed}}$ is the sum of the elastic energy associated to each tetrahedron.

4.2.3 Linear Elastic Force

Given the expression of the elastic energy, the force $\mathbf{F}_i$ applied to a vertex $\mathbf{P}_i$ is defined by:

$$\mathbf{F}_i = -\frac{\partial W_{Elastic}(\mathcal{M}_{\text{deformed}})}{\partial \mathbf{P}_i} = \sum_{T_j \in L(i)} -\frac{\partial W_{Elastic}(T_j)}{\partial \mathbf{P}_i}$$

where $L(i)$ is the set of tetrahedra adjacent to vertex $\mathbf{P}_i$.
Within the tetrahedron T_i, the force $\mathbf{F}_{T_i(j)}$ applied on vertex $\mathbf{P}_{T_j(j)}$ takes the following form:

$$\mathbf{F}_{T_j(j)} = \sum_{k=0}^{3} [\mathbf{K}^{\mathbf{T_i}}_{\mathbf{jk}}] \mathbf{P}^0_{T_i(k)} \mathbf{P}_{T_i(k)}$$

where $[\mathbf{K}^{\mathbf{T_i}}_{\mathbf{jk}}]$ are 3×3 stiffness matrices or tensors. Given a tetrahedron T_i and its four vertices $\mathbf{P}^0_{T_i(0)}, \mathbf{P}^0_{T_i(1)}, \mathbf{P}^0_{T_i(2)}, \mathbf{P}^0_{T_i(3)}$, we compute the six tensors $[\mathbf{K}^{\mathbf{T_i}}_{\mathbf{jk}}]$ as follows:

$$[K^{T_i}_{jk}] = \frac{1}{36V(T_i)} \left(\lambda_i \mathbf{M}_k^{T_i} (\mathbf{M}_j^{T_i})^T + \mu_i \mathbf{M}_j^{T_i} (\mathbf{M}_k^{T_i})^T + \mu_i (\mathbf{M}_j^{T_i} \mathbf{M}_k^{T_i}) \, [Id_{3\times 3}] \right) \tag{5}$$

with

$$[Id_{3\times 3}] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

In equation (5), $\mathbf{M}_k^{T_i}(\mathbf{M}_j^{T_i})^T$ corresponds to the tensorial product between vector $\mathbf{M}_k^{T_i}$ and vector $\mathbf{M}_j^{T_i}$. Moreover, equation (5) induces that $[\mathbf{K_{kj}^{T_i}}] = [\mathbf{K_{jk}^{T_i}}]^T$, which is equivalent to state that the rigidity matrix is symmetric.

It is important to notice that these stiffness matrices depend only on the material characteristics within a tetrahedron – described by the Lamé coefficients λ_i and μ_i – and the geometry of the tetrahedron T_i at its rest shape. For a given vertex $\mathbf{P}_i$, the elastic force $\mathbf{F}_i$ is the sum of all contributions $\mathbf{F}_{T_i(j)}$ from all adjacent tetrahedron T_i:

$$\mathbf{F}_i = [\mathbf{K_{ii}}]\mathbf{P}_i^0\mathbf{P}_i + \sum_{j\in N(\mathbf{P}_i)} [\mathbf{K_{ij}}]\mathbf{P}_j^0\mathbf{P}_j \tag{6}$$

where $[\mathbf{K_{ii}}]$ is the sum of tensors $[\mathbf{K_{ii}^{T_j}}]$ associated to the tetrahedra adjacent to $\mathbf{P}_i$, $[\mathbf{K_{ij}}]$ is the sum of tensors $[\mathbf{K_{ij}^{T_j}}]$ associated to the tetrahedra adjacent to edge (i,j) and $N(\mathbf{P}_i)$ is the list of $\mathbf{P}_i$ neighbors.

4.3 Data Structure

Given a tetrahedral mesh of a solid – in our case an anatomical structure – we designed a data structure that describes the set of vertices, edges and tetrahedra of the mesh. For each vertex, we store all adjacent tetrahedra, the current position $\mathbf{P}_i$, the rest position $\mathbf{P}_i^0$ and the tensor $[\mathbf{K_{ii}}]$. For each edge, we store the two adjacent vertices as well as the tensor $[\mathbf{K_{ij}}]$, as sketched in figure (2). Finally for each tetrahedron, we store a reference to its four vertices and its six edges as well as the Lamé coefficients λ_i, μ_i and the four vectors $\mathbf{M}_i$ defined in equation (10). See appendix (A) for more details.

4.4 Numerical Integration

We refer to the Newtonian differential equation (4) as the equation governing the motion of our linear elastic model. This equation is related to the differential equation found in continuum mechanics (Bathe, 1982):

$$\mathbf{M\ddot{U}} + \mathbf{C\dot{U}} + \mathbf{KU} = \mathbf{R} \tag{7}$$

Following finite elements theory, the mass matrix $\mathbf{M}$ and damping matrix $\mathbf{C}$ are sparse matrices that are related to the physical properties of each tetrahedron. In our case, we consider that $\mathbf{M}$ and $\mathbf{C}$ are diagonal matrices, i.e. that mass and damping effects are concentrated at vertices.

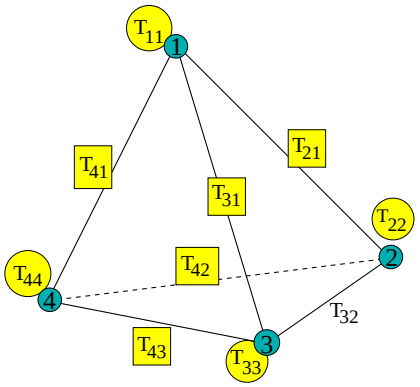


Figure 2: Representation of the data structure of a tensor-mass model. The tensors are stored at each edge and each vertex. The symmetry of the rigidity matrix enables to store only one tensor per edge.

This simplification called *mass-lumping* decouples the motion of all nodes and therefore allows writing equation (7) as the set of independent differential equations (4) for each vertex.

Furthermore, we choose an *explicit integration scheme* where the elastic force is estimated at time t in order to compute the vertex position at time $t+1$:

$$\left(\frac{m_i}{\Delta t^2} - \frac{\gamma_i}{2\Delta t}\right)\mathbf{P}_i^{t+1} = \mathbf{F}_i + \frac{2m_i}{\Delta t^2}\mathbf{P}_i^t - \left(\frac{m_i}{\Delta t^2} + \frac{\gamma_i}{2\Delta t}\right)\mathbf{P}_i^{t-1} \tag{8}$$

The key advantage of this explicit integration scheme is that no rigidity matrix inversion is required for updating each vertex. After modifying the mesh topology, equation (8) is used to update the vertex position without any additional computation besides the update of local tensors $[K_{ii}]$ (see section 5.5). On the other hand, the use of an implicit integration scheme would have entailed the inversion of the stiffness matrix every time the mesh topology would have been altered.

Implicit schemes are *unconditionally stable* which implies that large time steps may be used. However, these schemes require either to inverse a sparse matrix or to solve at each iteration a linear system of equations, as proposed in (Baraff and Witkin, 1998). On the other hand, explicit schemes are only *conditionally stable* and therefore they tend to converge more slowly than implicit schemes. In order to have computation time smaller than $20ms$, we have to use explicit schemes since the time for solving a sparse linear system is too large by at least one order of magnitude. For instance, computing one iteration step with the implicit scheme requires from 10 s to 1 mn depending on the stiffness material when using a preconditionned conjugated gradient method on a typical liver model. A discussion about explicit versus implicit schemes is also proposed in (Bro-Nielsen, 1998).

In this explicit scheme, the critical time step – i.e. the higher time step that can be used – is related to the highest eigenvalue of the rigidity matrix and to the local mass and damping

values (see (Bathe, 1982) for more details). To optimize the time step, we use constant values of the mass and damping values. Indeed, in (Bro-Nielsen, 1998), Bro-Nielsen proposed to use mass and damping values proportional to the volume of each tetrahedra. However this approach tends to create ill-conditioned iterative schemes (with low time-step) when dealing with meshes having tetrahedra of different size. In our approach, the choice of the time-step is only related to the stiffness of the soft tissue material and is independent of the elements size.

The speed of convergence is directly related to the time step and the damping values used in equation 8. Optimal damping values may be experimentally obtained in order to minimize the number of iterations to reach the equilibrium state. Furthermore, this number of iteration is inversely proportional to the time step Δt. Finally, we use a fourth order Runge-Kutta method (Press et al., 1991) for discretizing the time domain instead of the Euler method described in equation 8. This method requires to evaluate four times the forces applied to $P_i(t)$ in order to compute the next position $P_i(t+1)$. In spite of this extra computational cost, our conclusion is that the Runge-Kutta method allows using larger time steps – about ten times larger – than the Euler method.thus leading to a speed-up factor of about two. This is particularly interesting when deforming stiff material which requires higher frequency computations for realistic haptic feedback.

4.5 Simulation of Cutting and Tearing

One of the basic tasks in surgery simulation consists in cutting and tearing soft tissue. With the dynamic linear elastic model, these tasks can be achieved in real-time.

We simulate the action of an electric scalpel – a bipolar cautery instrument – on soft tissue by successively removing tetrahedra at places where the instrument is in contact with the anatomical model. This approach implies that for realistic simulation, the tetrahedra must be relatively small at the regions where the cutting may occur. Furthermore, in order to keep the mesh conformal, additional tetrahedra may be automatically removed after checking the local vertex and edge adjacency. Rather than removing tetrahedra, more sophisticated cutting operations could be performed, for instance by locally remeshing around the contact zone.

When a collision between the instrument and a tetrahedron is detected, the local deformation tensors associated with the tetrahedron are computed and then subtracted from the current deformation tensors at the edges and vertices of the tetrahedron. Since the update of the tensors is only local, this is performed in a very efficient way. For instance, when removing the tetrahedron T_i, the 6 edge tensors $[K_{jk}]$ and the 4 vertex tensors $[K_{jj}]$ are updated by removing the tensors $[K_{jk}^{T_i}]$ and $[K_{jj}^{T_i}]$ associated with T_i. More precisely, the following ten operations are

performed :

$$[K_{jj}] = [K_{jj}] - [K_{jj}^{T_i}] \qquad [K_{jk}] = [K_{jk}] - [K_{jk}^{T_i}]$$

Finally, we update the list of displayed triangles if a tetrahedron is located at the volumetric model boundary. By locally updating tensors, the tissue has exactly the same behavior as if we had removed the corresponding tetrahedra at its rest position. Because of the volumetric continuity of finite element modeling, the deformation of the tissue is continuous and appears very natural during the cutting.

In addition to cutting, the tearing of soft tissue can be simulated. The tearing occurs at places where normal stress and shearing are too high. The basic algorithm is to compute a local deformation measure for each tetrahedron. If this criterion is greater than a threshold, the tetrahedron is removed and the tensors are locally updated. We have implemented three geometric criteria for detecting highly deformed tetrahedra. These criteria are the relative variation of volume, the mean relative elongation of the six edges and the maximum relative elongation of the edges.

4.6 Spring-Mass Model versus Tensor-Mass Model

In a classical approach, a vertex $\mathbf{P}_i$ in a spring-mass system, is submitted to an elastic force:

$$\mathbf{F}_i = \sum_{j \in N(\mathbf{P}_i)} k_{ij}(\|\mathbf{P}_i\mathbf{P}_j\| - l_{ij}^0)\frac{\mathbf{P}_i\mathbf{P}_j}{\|\mathbf{P}_i\mathbf{P}_j\|} \tag{9}$$

where $N(\mathbf{P}_i)$ is the set of vertices $\mathbf{P}_j$ adjacent to $\mathbf{P}_i$, k_{ij} is the stiffness coefficient between vertices $\mathbf{P}_i$ and $\mathbf{P}_j$, l_{ij}^0 is the rest length between $\mathbf{P}_i$ and $\mathbf{P}_j$.
We have shown that every vertex in our dynamic linear elastic model is submitted to the force of equation (6). Because of the similarity between the two approaches, we have coined the term **tensor-mass** in order to describe this dynamic linear elastic model.

By comparing equations (6) and (9), it is clear that both dynamic models have the same computational complexity which is linear in the number of edges. In practice, we were able to reach an update frequency of 40 Hz [4] with a tetrahedral mesh having 760 vertices and about 4000 edges and similar results were obtained for a spring-mass system. The tensor-mass model does not require any square root evaluation, but slightly more information must be pre-computed than for spring-mass systems.

However, both approaches substantially differ in terms of biomechanical modeling. Spring-mass systems constitute a discrete representation of an object and their behavior strongly depend on the topology of the spring network. When a spring is removed or added it may change

[4] These computing times where obtained on a 233 Mhz DEC Alphastation.

drastically the elastic behavior of the whole system. Conversely, our finite elastic model is a continuous representation of the object and its behavior is independent of the mesh topology (it mostly depends on the mesh resolution). This implies that when the mesh is cut, continuous and natural behavior of the tissue is simulated.

Because all biomechanical data related to biological soft tissue are formulated as parameters of the continuum mechanics (such as Young modulus or Poisson coefficients), it is *a priori* difficult to model realistic soft tissue deformations with a spring-mass system. However, several authors (Jean Louchet, 1995; Deussen et al., 1995) have developed genetic or simulated annealing algorithms to identify springs parameters (stiffness and damping) from a set of known deformations of an object.

Finally as previously mentioned, the linear elastic model is only valid for small displacements. For instance, if a rigid transformation is applied to the rest shape $\mathcal{M}_{initial}$, then the forces applied to all vertices will not be null. On the opposite, a spring-mass model under the same displacement would not deform, since the length of the springs are preserved under a rigid transformation. The difference between these three soft tissue models is summarized in Table 1.

	Pre-computed	Tensor-Mass	Spring-Mass
Computational efficiency	+++	+	+
Biomechanical Realism	+	+	-
Cutting simulation	-	++	+
Large displacements	-	-	+

Table 1: Comparison between the three soft tissue models: pre-computed quasi-static, tensor-mass and spring-mass models.

4.7 Examples

We present two examples of deformations generated by a tensor-mass model. Figure (3) shows a volumetric plate being cut while its four corners are fixed. The cutting consists in selecting a set of tetrahedra interactively with the mouse (figure 3 (a)) and then updating the local tensors surrounding the removed tetrahedra. As more and more tetrahedra are removed, one can see that the plate is deforming. This is due to the fact that the overall stiffness of the material is modified by the cutting operation.

In figure (4), we show a cylinder that is deformed under a gravity force. Based on the relative change of volume, the tearing algorithm removes automatically the tetrahedra where the relative change of volume becomes greater than a given threshold, thus exhibiting a fracture.

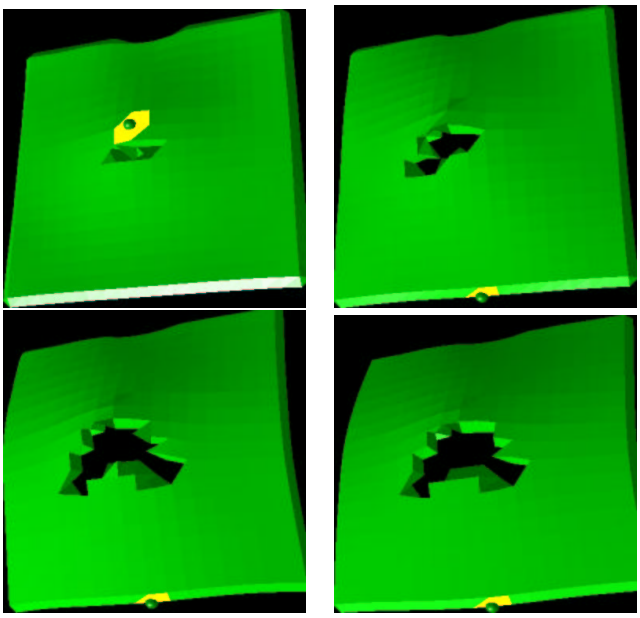


Figure 3: Cutting of a plate by removing tetrahedra. The plate deforms itself during the cutting operations because each operation modifies the stiffness matrix. The plate model consists of 759 vertices and 2212 tetrahedra.

The cylinder is composed of 248 vertices and 889 tetrahedra.

5 Hybrid elastic model for surgery simulation

5.1 Motivations

We have previously described two *linear elastic models* that have the following properties:

1. The *quasi-static* pre-computed elastic model is extremely efficient but does not allow topology change (cutting, tearing) (see section 4).
2. The *dynamic* elastic model (or tensor-mass model) requires more computation but authorizes topology change (see section 5).

In this section, we propose to combine these two approaches in order to optimize the trade-off between computation time and visual realism of the simulation.

The key idea consists in a separate modeling of two different types of anatomical structures that usually appear in a surgical simulation:

- The anatomical structures that are the target of the surgical procedure. On these structures, tearing and cutting need to be simulated. In many cases, they correspond to patho-

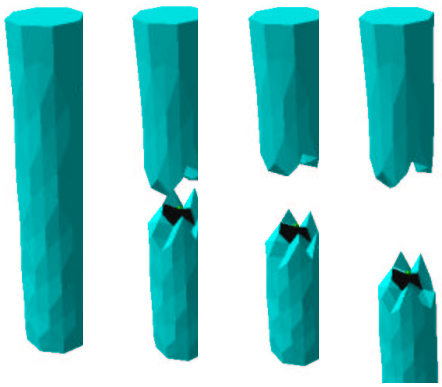


Figure 4: Fracture of a cylinder under the action of a gravity force.

logical structures and only represent a small subset of the anatomy that needs to be visualized during the simulation.

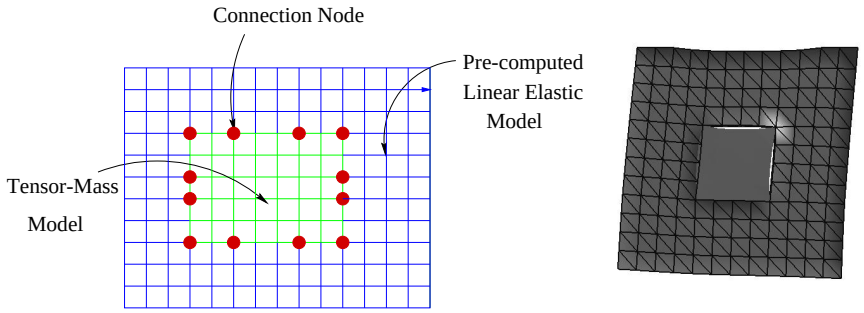
- The anatomical structures that only need to be visualized or eventually deformed. They contribute to the visual realism of the simulation but are not submitted to any surgical action.

The former type of anatomical structures are good candidates for being modeled as tensor-mass models whereas the latter should be modeled with a pre-computed linear model. However, this approach is really optimal when different parts of the same anatomical structure can be modeled either as a pre-computed elastic model or a tensor-mass model. This is a way to decrease the number of tensor-mass elements to its minimum and therefore to increase the interactivity of the overall simulation.

In the next sections, we describe how to connect these two linear elastic models together in order to represent a global deformable object. We have called *hybrid elastic model* this combined representation.

5.2 Hybrid elastic model

Let's consider a hybrid elastic model $\mathcal{M}_{hybrid}$ composed of two different types of elements. We denote as $\mathcal{M}_{dynamic}$ the set of tensor-mass elements and we denote as $\mathcal{M}_{quasi-static}$ the set of pre-computed linear elastic elements. Therefore a hybrid elastic model may be composed of several pieces of tensor-mass models each corresponding to a structure that plays a role in the surgical procedure. The model $\mathcal{M}_{dynamic}$ is connected to $\mathcal{M}_{quasi-static}$ by a set of common vertices or *connection nodes*. These connection nodes define additional boundary conditions for each model. As seen in figure (5.2), the two models may not be completely connected along their common boundaries. Actually, a way to reduce the number of tensor-mass elements, is



(a) (b)

Figure 5: (a) Definition of the connection nodes in a hybrid elastic model; (b) Hybrid elastic model with eight connection nodes.

to associate a fine pre-computed elastic model with a coarse tensor-mass model. As shown in figure (5.2-b) , this incomplete connection entails some visual artifacts due to the non-continuity between the two parts. However, if the connection part between the two elastic models is not an important visual cue, a different mesh resolution can be used.

Since both linear elastic models follow the same physical law, the combination of these two models should behave exactly as a global linear elastic model. To achieve this goal, the additional boundary conditions imposed at the connection nodes must be consistent in terms of forces and displacements for both models.

Boundary Conditions	Quasi-static Pre-computed model	Tensor-Mass Model
Applied Forces	No	Yes
Constrained displacement	Yes	Yes

Table 2: Natural boundary conditions for the quasi-static and dynamic linear elastic models.

We have shortly summarized the "natural" boundary conditions for both elastic models in Table (2). Since the pre-computed model only supports displacement boundary conditions, $\mathcal{M}_{dynamic}$ will constraint the displacements of the connection nodes whereas $\mathcal{M}_{quasi-static}$ applies a force on the connection nodes.

Figure (6) summarizes the computation loop of a hybrid model. The pre-computed elastic model $\mathcal{M}_{quasi-static}$ is updated based on the imposed displacements on its boundary. These displacements arise from the user interaction – action of a surgical tool for instance – and from the motion of the connection nodes associated with a tensor-mass mesh. At this stage, the

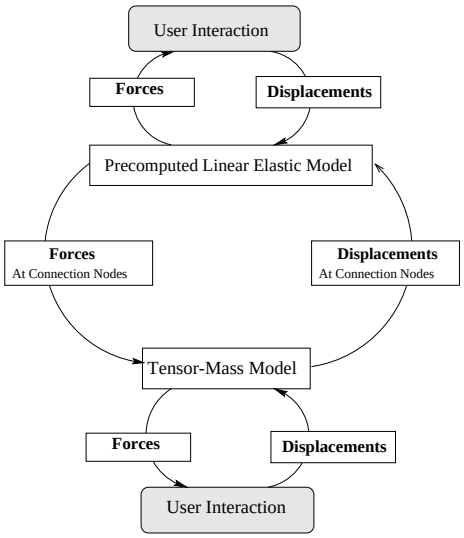


Figure 6: Interaction loop for a hybrid elastic model. Both models are updated alternatively while allowing for user interaction.

displacement of every surface node of $\mathcal{M}_{quasi-static}$ is computed as well as the resulting force at each constraint node - including the connection nodes.

After updating $\mathcal{M}_{\mathrm{quasi-static}}$, we update $\mathcal{M}_{\mathrm{dynamic}}$ based on the forces applied on the connection nodes and the displacements imposed by the user interaction. New positions and forces are computed for every node including the connection nodes. Again, the updated position of the connection nodes constitute new displacement constraints for $\mathcal{M}_{\mathrm{quasi-static}}$.

In figure (7), we present an example of a hybrid cylinder model undergoing a deformation under the action of a gravity force. The figure shows the different steps of the deformation process and, on the right, the steady state reached by the model. At the equilibrium the forces applied to all connection nodes are null and the displacement vector stabilizes to a constant value. In this example, both quasi-static and dynamic models have the same elastic properties since their governing equations use the same Lamé coefficients. We have then verified that the steady state reached by the hybrid model is the same as the steady state that would have reached a single quasi-static or dynamic elastic model.

5.3 Surgery simulation on hybrid elastic models

To demonstrate the efficiency of our approach for performing surgery simulation, we have chosen to simulate an hepatectomy. An hepatectomy consists in removing one of the eight anatomical segments – Couinaud segments (Couinaud, 1957) – of a liver. In this example the segment number six has to be removed. The shape of the liver was segmented using a deformable

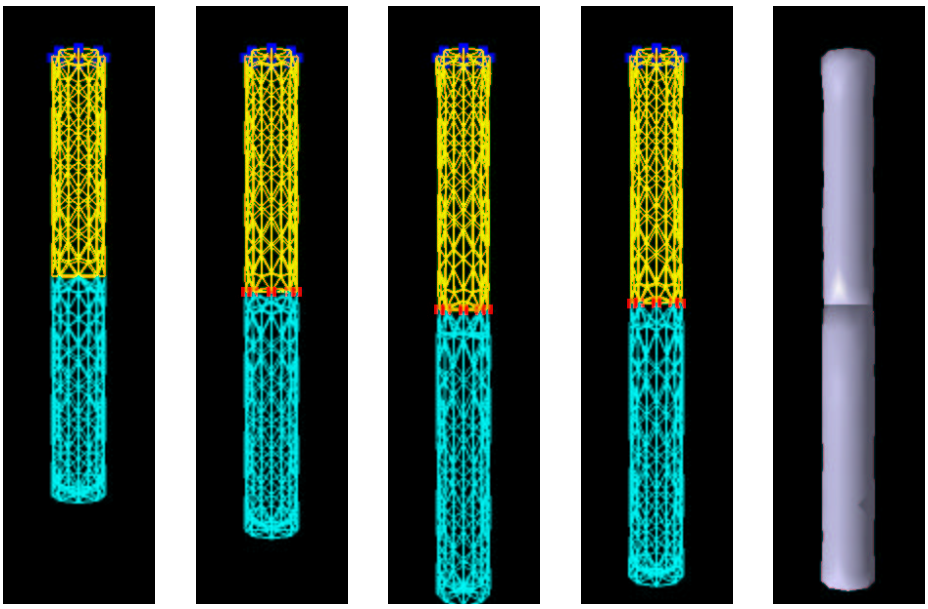


Figure 7: Deformation of a hybrid elastic model under a gravity force: the upper cylinder consists of a pre-computed linear elastic model whereas the lower part is a tensor-mass model. The leftmost figure corresponds to the initial position of the mesh and the rightmost figure to the steady state.

simplex mesh (Montagnat and Delingette, 1997) on a helicoïdal CT-scan image of the abdomen. This segmentation generates a triangulated surface. Then a tetrahedral mesh has been created from this triangulation using the Simail (Simail: product of Simulog S.A. - 1, rue James Joule - 78286 Guyancourt Cedex - France,) commercial software. The resulting volumetric mesh is composed of 1537 vertices and 7039 tetrahedra – see figure (8). About 18% (280 vertices and 1260 tetrahedra) of the liver hybrid mesh is modeled as tensor-mass system and the remaining is defined as a pre-computed linear elastic model.

The surgery simulator consists of two force-feedback systems simulating the elongated surgical instruments used in laparoscopy. These force-feedback devices are plugged into a PC computer that is linked to a powerful computer (SGI Onyx2 Infinite Reality with 2 processors) via an ethernet network. On that computer, the collision detection is performed as well as the animation of the hybrid elastic model. The collision detection algorithm and some other details on the hardware architecture of the simulator can be found in (Cotin et al., 1999). The link between the two computers is only used to transmit force or position information at a high speed.

In figure 9, we show different stages of the hepatectomy simulation. With these two virtual instruments, it is possible to push any part of the liver (either represented with a quasi-static or a dynamic model) and to cut any elements of the dynamic model located in the bottom part of

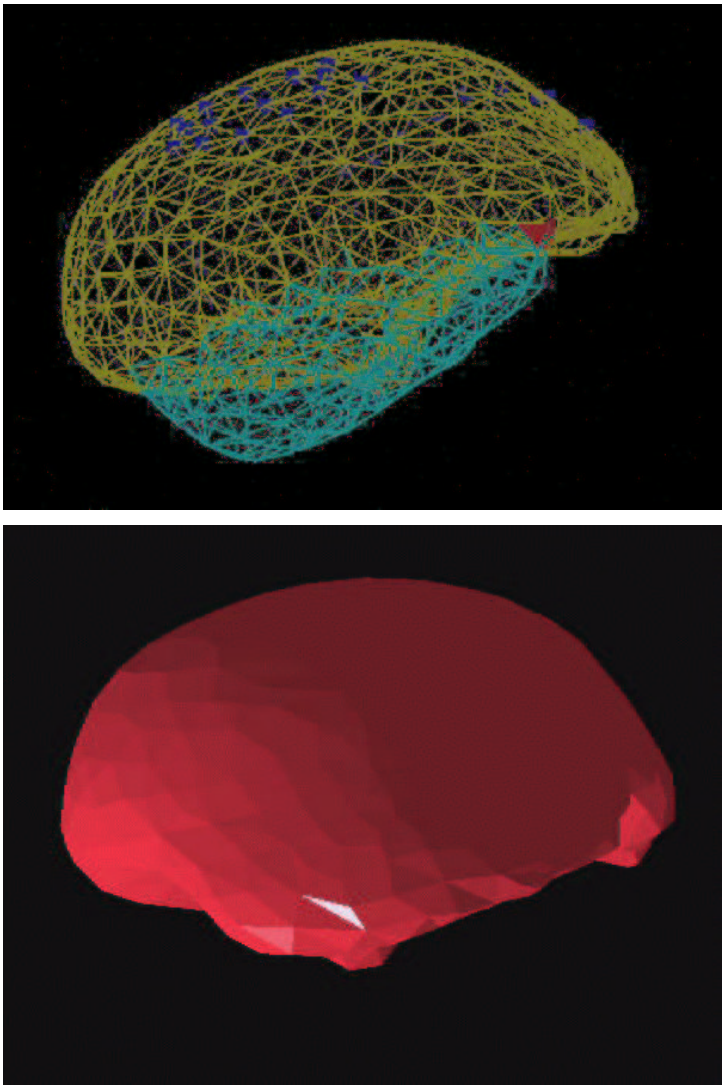


Figure 8: **Top**: the hybrid liver model seen in wireframe. The upper mesh corresponds to the pre-computed quasi-static elastic model whereas the bottom mesh corresponds to the tensor-mass model. **Bottom** : the hybrid liver model seen in flat shading. The connection nodes ensure the visual continuity between the two elastic models.

the screen. The first six pictures show the deformation of the model when the tool collides with the dynamic model. Since both models have the same elastic characteristics, it is not possible to visually distinguish the interface between the two different elastic models. In the current implementation, once the tool collides with the mesh, it stays connected to the same triangle during the collision, thus corresponding to a surface with infinite friction.

The last six pictures show the cutting of the liver segment by removing additional tetrahedra. The cutting occurs for the tetrahedron being collided by the tool. One can notice that each part of the hybrid model naturally deforms itself during the resection simulation.

6 Conclusion

We have presented three different soft tissue models based on linear elasticity. The first model, introduced in (Cotin et al., 1999), is extremely efficient but does not allow the simulation of cutting. The second model called "tensor-mass model" allows to simulate the dynamics of soft tissue, similarly to spring-mass models. We believe that tensor-mass models are a convincing alternative to using spring-mass models because of their simple implementation and because they are based on a continuous representation of the tissue. We have shown that the simulation of tearing and cutting procedures can be easily performed with these models due to their compact data structure.

Finally, we have proposed hybrid elastic models that combine both previously described elastic models therefore enabling to cut and deform large anatomical structures. With these algorithms, we have demonstrated that it is possible to use meshes having more than 8000 tetrahedra for simulating surgical gestures including deformations and cutting.

This approach has mainly two limitations. First, the zones where a cutting gesture can take place, are limited by the workstation computing power of workstation, the number of nodes of the soft tissue to be cut, and the stiffness of that soft tissue. In our experiment, we were able to use tensor-mass models large enough to simulate a realistic surgical gesture. However, for stiff materials, using explicit integration schemes may not be suitable for a proper simulation. To obtain more efficient algorithm, we are investigating several speed-ud algorithms based on parallel-computation and mesh adaptation(Lombardo et al., 1999; Picinbonno and Lombardo, 1999).

The second limitation lies in the choice of the biomechanical behavior. Linear elasticity is only valid for small displacements and small deformations. With a linear elastic model, as the surgeon pushes his instrument along a given direction, the trajectory of mesh vertices follow a straight line. To simulate gestures involving rotations of anatomical structures (such as rotating a liver lobe), it is important to take into account large displacements. Also, since most biological

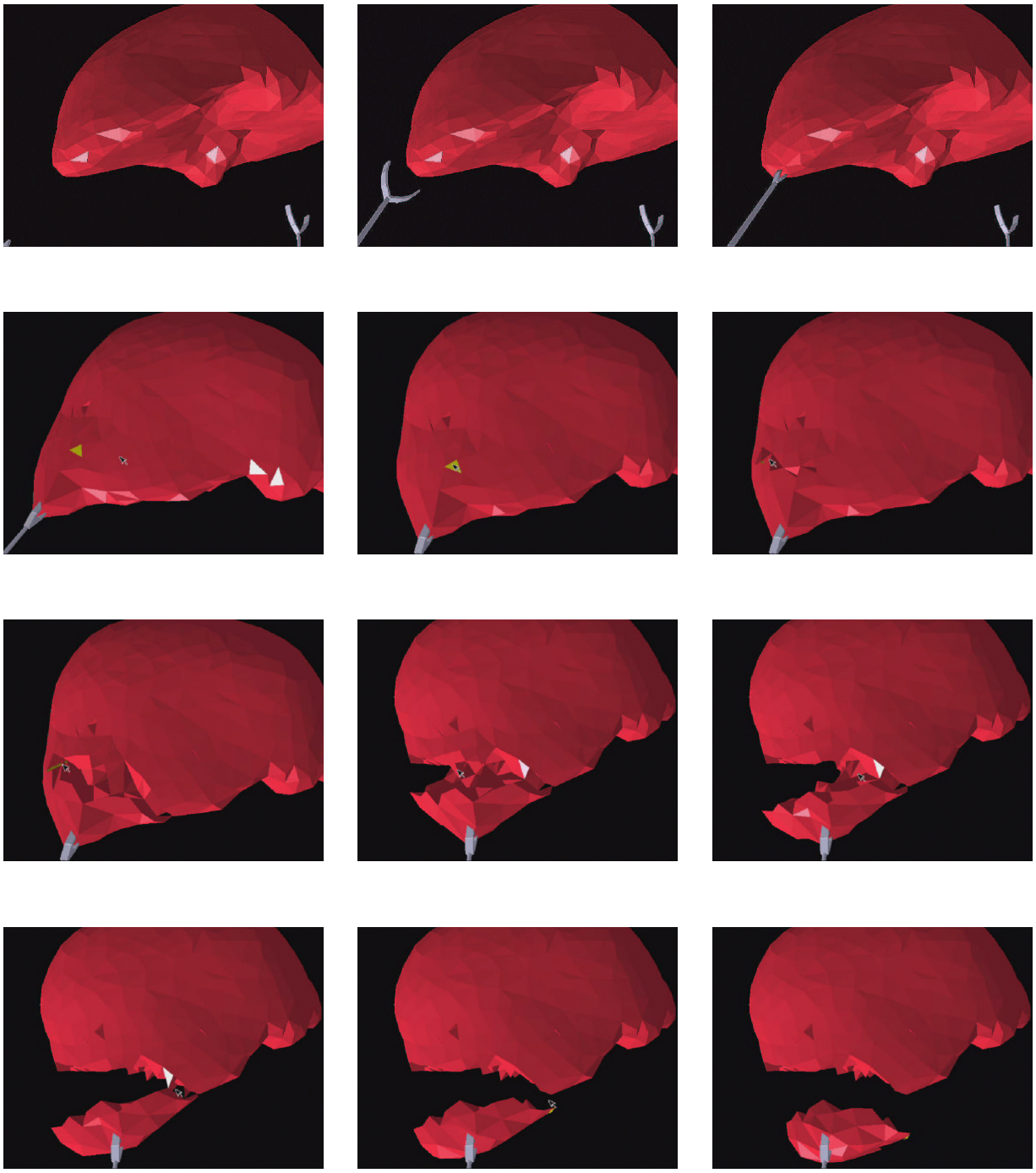


Figure 9: Deformation and cutting of the hybrid elastic model.

tissues behaves as non-linear materials, the support for large deformations is also needed for a better visual and haptic feedback. Currently, we are developing soft tissue models with an anisotropic and non-linear elastic behavior under large displacements.

Acknowledgments

This project was partially funded by a contract with IRCAD with the collaboration of professor J. Marescaux, director of IRCAD and the help of Y. Russier, J.M. Clément and L. Soler. The authors are grateful for their collaboration in this project and their valuable advice. The author also thank J-C. Lombardo, G. Picinbonno and all the members of the INRIA incentive action AISIM for stimulating discussions and G. Kahn for his permanent interest and support throughout this research.

A Displacement inside a tetrahedron

Given a tetrahedron T_i and its four vertices $\mathbf{P}^0_{T_i(0)}, \mathbf{P}^0_{T_i(1)}, \mathbf{P}^0_{T_i(2)}, \mathbf{P}^0_{T_i(3)}$, one computes the displacement inside T_i through a linear interpolation of the four displacement $\mathbf{U}_{T_i(j)}$.

Given a point $\mathbf{x}$ inside T_i, its barycentric coordinates $\alpha_j^{T_i}$ of $\mathbf{U}_{T_i(j)}$ can be written as :

$$\alpha_j^{T_i}(\mathbf{x}) = -\frac{\mathbf{M}_j^{T_i}.(\mathbf{x} - \mathbf{P}^0_{T_i(j+1)})}{6V(T_i)}$$

where $V(T_i)$ is the volume of tetrahedron T_i and $\mathbf{M}_j^{T_i}$ is the vector:

$$\begin{aligned}
\mathbf{M}_j^{T_i} &= \left[M_j^x\ M_j^y\ M_j^z\right]^T && (10)\\
&= \mathbf{P}^0_{T_i(j+1)} \wedge \mathbf{P}^0_{T_i(j+2)} + \mathbf{P}^0_{T_i(j+2)} \wedge \mathbf{P}^0_{T_i(j+3)} + && (11)\\
&\quad \mathbf{P}^0_{T_i(j+3)} \wedge \mathbf{P}^0_{T_i(j+1)} && (12)
\end{aligned}$$

$\mathbf{M}_j^{T_i}$ has a simple geometrical interpretation : $\mathbf{M}_j^{T_i}$ is directed along the outer normal vector of triangle opposite to $\mathbf{P}^0_{T_i(j)}$ and $\|\mathbf{M}_j^{T_i}\|$ represents twice the area of the triangle opposite to $\mathbf{P}^0_{T_i(j)}$.
The displacement $\mathbf{U}(\mathbf{x})$ is then written as:

$$\mathbf{U}(\mathbf{x}) = \sum_{j=0}^{3} \alpha_j^{T_i} \mathbf{U}_{T_i(j)} \tag{13}$$

B Elastic energy of a tetrahedron

The elastic energy of equation 1 can also be written as:

$$W_{Elastic} = \int_{\mathcal{M}_{\text{initial}}} \left(\frac{\lambda}{2} (div\ \mathbf{U})^2 + \mu\ tr\ (\nabla \mathbf{U}^T \nabla \mathbf{U}) - \frac{\mu}{2} \|curl\ \mathbf{U}\|^2 \right) dx\ dy\ dz$$

Given the expression of the displacement vector $\mathbf{U}(\mathbf{x})$ inside a tetrahedron T_i (see equation 13), we get the expression of the operator divergence $div\mathbf{U}$, rotational $curl\mathbf{U}$ and the gradient norm $tr\ (\nabla \mathbf{U}^T \nabla \mathbf{U})$. For instance, the divergence operator $div\mathbf{U}$ can be derived as follows :

$$\begin{aligned}
div\mathbf{U} &= \frac{\partial U_x}{\partial x} + \frac{\partial U_y}{\partial y} + \frac{\partial U_z}{\partial z}\\
&= \sum_{j=0}^{3} \left(\frac{\partial \alpha_j^{T_i}(\mathbf{x}) \mathbf{U}^x_{T_i(j)}}{\partial x} + \frac{\partial \alpha_j^{T_i}(\mathbf{x}) \mathbf{U}^y_{T_i(j)}}{\partial y} + \frac{\partial \alpha_j^{T_i}(\mathbf{x}) \mathbf{U}^z_{T_i(j)}}{\partial z} \right)\\
&= \sum_{j=0}^{3} \frac{\partial \alpha_j^{T_i}(\mathbf{x})}{\partial \mathbf{x}} \cdot \mathbf{U}_{T_i(j)}\\
&= \sum_{j=0}^{3} \frac{-\mathbf{M}_j.\mathbf{U}_{T_i(j)}}{6V(T_i)}
\end{aligned}$$

In a similar way, the rotation and gradient norm operator are evaluated as follows :

$$curl\,\mathbf{U} = \sum_{j=0}^{3} \frac{-\mathbf{M}_j \wedge \mathbf{U}_{T_i(j)}}{6V(T_i)}$$

$$\begin{aligned} tr\,(\nabla \mathbf{U}^T \nabla \mathbf{U}) = & \left\| \sum_{j=0}^{3} \frac{(\mathbf{M}_j.\mathbf{x})\mathbf{U}_{T_i(j)}}{6V(T_i)} \right\|^2 \\ & + \left\| \sum_{j=0}^{3} \frac{(\mathbf{M}_j.\mathbf{y})\mathbf{U}_{T_i(j)}}{6V(T_i)} \right\|^2 \\ & + \left\| \sum_{j=0}^{3} \frac{(\mathbf{M}_j.\mathbf{z})\mathbf{U}_{T_i(j)}}{6V(T_i)} \right\|^2 \end{aligned}$$

Because the strain tensor $E(\mathbf{x})$ is constant across T_i (and consequently the elastic energy), to obtain the elastic energy of tetrahedron T_i, we multiply the constant elementary elastic energy by the volume of T_i :

$$\begin{aligned} & W_{\mathrm{E}lastic}(T_i) = \\ & \quad V(T_i) \left(\frac{\lambda_i}{2} (\mathrm{d}iv\,\mathbf{U})^2 + \mu_i tr\,(\nabla \mathbf{U}^T \nabla \mathbf{U}) - \frac{\mu_i}{2} \|curl\,\mathbf{U}\|^2 \right) \end{aligned}$$

References

Ayache, N., Cotin, S., and Delingette, H. (1997). Surgery simulation with visual and haptic feedback. In Shirai, Y. and Hirose, S., editors, *8th International Symposium on Robotics Research*, pages 311–316. Springer, Japan.

Ayache, N., Cotin, S., Delingette, H., Clement, J.-M., Marescaux, J., and Nord, M. (1998). Simulation of endoscopic surgery. *Journal of Minimally Invasive Therapy and Allied Technologies (MITAT)*, 7(2):71–77.

Bainville, E. (1996). *Modélisation géométrique et dynamique d'un geste chirurgical.* PhD thesis, Université Joseph Fourier.

Bainville, E., Chaffanjon, P., and Cinquin, P. (1995). Computer generated visual assistance during retroperitoneoscopy. *Computers in biology and medecine*, 2(25):165–171.

Baraff, D. and Witkin, A. (1998). Large steps in cloth simulation. In ACM, editor, *Computer Graphics (SIGGRAPH'98)*, pages 43–54, Orlando (USA).

Bathe, K.-L. (1982). *Finite Element Procedures in Engineering Analysis.* Prentice-Hall.

Baumann, R. and Glauser, D. (1996). Force Feedback for Virtual Reality based Minimally Invasive Surgery Simulator. In *Medecine Meets Virtual Reality*, volume 4, San Diego, CA.

Bro-Nielsen, M. (1996). *Medical Image Registration and Surgery Simulation.* PhD thesis, IMM Technical University of Denmark, Lingby, Denmark. IMM-PHD-1996-25.

Bro-Nielsen, M. (1998). Finite element modeling in surgery simulation. *Proceedings of the IEEE : Special Issue on Surgery Simulation*, pages 490–503.

Bro-Nielsen, M. and Cotin, S. (1996). Real-time Volumetric Deformable Models for Surgery Simulation using Finite Elements and Condensation. In *Proceedings of Eurographics'96 - Computer Graphics Forum*, volume 15, pages 57–66.

Chen, D. T. and Zeltzer, D. (1992). Pump it up: Computer animation of a biomechanically based model of muscle using the finite element method. In Catmull, E. E., editor, *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 89–98.

Ciarlet, P. G. (1987). *Mathematical elasticity Vol. 1: Three-dimensional elasticity.* , Amsterdam. ISBN 0-444-70259-8.

Cotin, S. (1997). *Modèles anatomiques déformables en temps-réel : Application à la simulation de chirurgie avec retour d'effort.* PhD thesis, Université de Nice-Sophia Antipolis.

Cotin, S., Delingette, H., and Ayache, N. (1999). Real-time elastic deformations of soft tissues for surgery simulation. *IEEE Transactions On Visualization and Computer Graphics*, 5(1):62–73.

Cotin, S., Delingette, H., Clément, J.-M., Tassetti, V., Marescaux, J., and Ayache, N. (1996). Geometric and Physical Representations for a Simulator of Hepatic Surgery. In *Proceedings of Medecine Meets Virtual Reality IV*, pages 139–151. IOS Press.

Couinaud (1957). *Le foie, études anatomiques et chirurgicales.* Masson.

Delingette, H. (1998). Towards realistic soft tissue modeling in medical simulation. *Proceedings of the IEEE : Special Issue on Surgery Simulation*, pages 512–523.

Deussen, O., Kobbelt, L., and Tucke, P. (1995). Using simulated annealing to obtain a good approximations of deformable bodies. In *Proc. Eurographics Workshop on Animation and Simulation.*

Fung, Y. C. (1993). *Biomechanics - Mechanical Properties of Living Tissues.* Springer-Verlag, second edition.

Gibson, S., Samosky, J., Mor, A., Fyock, C., Grimson, E., Kanade, T., Kikinis, R., Lauer, H., and McKenzie, N. (1997). Simulating arthroscopic knee surgery using volumetric object representations, real-time volume rendering and haptic feedback . In Troccaz, J., Grimson, E., and Mosges, R., editors, *Proceedings of the First Joint Conference CVRMed-MRCAS'97*, volume 1205 of *Lecture Notes in Computer Science*, pages 369–378.

Gourret, J. P., Magnenat-Thalmann, N., and Thalmann, D. (1989). Simulation of Object and Human Skin Deformations in a Grasping Task. In *Computer Graphics (SIGGRAPH'89)*, volume 23 No 3, pages 21–31, Boston, MA, USA.

Jean Louchet, Xavier Provot, D. C. (1995). Evolutionary identification of cloth animation model. In *Workshop on Computer Animation and Simulation (Eurographics'95)*, pages 44–54.

Kaiss, M. and Le Tallec, P. (1996). La Modélisation numérique du contact œil-trépan. *Revue Européenne des Eléments Finis*, 5(3):375–408.

Kuehnapfel, U. and Neisius, B. (1993). CAD-Based Graphical Computer Simulation in Endoscopic Surgery. *End. Surg.*, 1:181–184.

Lombardo, J.-C., Cani, M.-P., and Neyret, F. (1999). Real-time Collision Detection for Virtual Surgery . In *Computer Animation*, Geneva - Switzerland.

Maurel, W., Wu, Y., and Thalmann, N. M. T. D. (1998). *Biomechanical Models for Soft Tissue Simulation.* ESPRIT Basic Research Series. Springer-Verlag.

Meseure, P. and Chaillou, C. (1997). Deformable Body Simulation with Adaptative Subdivision and Cuttings. In *Proceedings of the WSCG'97*, pages 361–370.

Montagnat, J. and Delingette, H. (1997). Volumetric Medical Images Segmentation using Shape Constrained Deformable Models. In *Computer Vision, Virtual Reality and Robotics in Medicine*, pages 13–22.

Picinbonno, G. and Lombardo, J.-C. (1999). Extrapolation : a solution for force feedback. In *Workshop on Virtual Reality and Prototyping (Laval Virtual'99)*, Laval (France).

Platt, J. C. and Barr, A. H. (1988). Constraint Methods for Flexible Models. In *Computer Graphics (SIGGRAPH'88)*, volume 22 No 4, pages 279–288.

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. (1991). *Numerical Recipes in C.* Cambridge Press.

Simail: product of Simulog S.A. - 1, rue James Joule - 78286 Guyancourt Cedex - France.

Song, G. J. and Reddy, N. P. (1995). Tissue Cutting In Virtual Environment. In *Medecine Meets Virtual Reality IV*, pages 359–364. IOS Press.

Speeter, T. H. (1992). Three Dimensional Finite Element Analysis of Elastic Continua for Tactile Sensing. *The International Journal of Robotics Research*, 11 No 1:1–19.

Szekely, G., ans R. Hutter, C. B., Rhomberg, A., and Schmid, P. (1998). Modelling of soft tissue deformation for laparoscopic surgery simulation. In Wells, Colchester, and Delp, editors, *Medical Image Computing and Computer-Assisted Intervention (MICCAI'98)*, number 1496 in Lecture Notes in Computer Science, pages 550–561, Cambridge (USA). Springer.

Terzopoulos, D., Platt, J., Barr, A., and Fleischer, K. (1987). Elastically deformable models. In Stone, M. C., editor, *Computer Graphics (SIGGRAPH '87 Proceedings)*, volume 21, pages 205–214.

Waters, K. (1992). A physical model of facial tissue and muscle articulation derived from computer tomography data. In *Visualization in Biomedical Computing (VBC'92)*, volume 574, Chappel Hill, NC.

Zienkiewicz, O. (1977). *The finite element method.* McGraw-Hill, London, 3 edition.

List of Figure Legends

- **Figure 1:** Deformation of plate represented as a pre-computed linear elastic model : (left) initial position (right) deformed position after imposing a displacement at the contact point with the stick.
- **Figure 2:** Representation of the data structure of a tensor-mass model. The tensors are stored at each edge and each vertex. The symmetry of the rigidity matrix enables to store only one tensor per edge.
- **Figure 3:** Cutting of a plate by removing tetrahedra. The plate deforms itself during the cutting operations because each operation modifies the stiffness matrix. The plate model consists of 759 vertices and 2212 tetrahedra.
- **Figure 4:** Fracture of a cylinder under the action of a gravity force.
- **Figure 5:** (a) Definition of the connection nodes in a hybrid elastic model; (b) Hybrid elastic model with eight connection nodes.
- **Figure 6:** Interaction loop for a hybrid elastic model. Both models are updated alternatively while allowing for user interaction.
- **Figure 7:** Deformation of a hybrid elastic model under a gravity force: the upper cylinder consists of a pre-computed linear elastic model whereas the lower part is a tensor-mass model. The leftmost figure corresponds to the initial position of the mesh and the rightmost figure to the steady state.
- **Figure 8:** **Top**: the hybrid liver model seen in wireframe. The upper mesh corresponds to the pre-computed quasi-static elastic model whereas the bottom mesh corresponds to the tensor-mass model. **Bottom** : the hybrid liver model seen in flat shading. The connection nodes ensure the visual continuity between the two elastic models.
- **Figure 9:** Deformation and cutting of the hybrid elastic model.